Final Project

Present Game : NINJA RUN

By

Team: SUPOP__TEAM__A
Member:
Thayakorn Fu 5931027321
Kittikarn Tuntiwiwut 5931007821

Programming Methodology
2110215 (2017/1)

Project Ninja Run

1. Problem Statement : Ninja Run Game

Ninja Run game is a game which ninja run and avoid many obstruct that appear on the screen. How long ninja can run , the score will increase. Ninja can take damage in 5 times. So ninja must collect items , Game has two types of items. The first is sushi can heal, Second is scroll which collect in 3 times ninja will can use ultimate skill.

The game has 6 states. The intro state is the first one shown up when player open game to present my team.

Figure 1 : The intro state of the application.

At intro state player can press ENTER to skip / wait a little time to switch to menu state.

The menu state is the second one to let player select option .

Figure 2: the menu state of application.

At this menu state , player can .

- Press Enter to select option.
- Up/Down arrow to move option.

When player press Enter on the Start option. The program's window switches to the playing state

The playing state has 4 sub state. The start game state is the first one to show how to control and what items that can collect them.

Figure 3: the start game state of playing state .

At start game state , player can.

- Press any keys to start game.

  When player press any key the game will start.

Figure 4 :the game play state of playing state.

Figure 5: Ultimate skill alert

Figure 6: Use ultimate skill

At play game state, player can.

- Press UP to jump
- Press RIGHT to warp (When use it will caused Chakra)
- Press LEFT to use ultimate skill when ultimate skill alert
- Press DOWN to sprint
- Press SPACE BAR to attack
- Press ESC to Pause

When player press ESC , game is paused and display pause state.

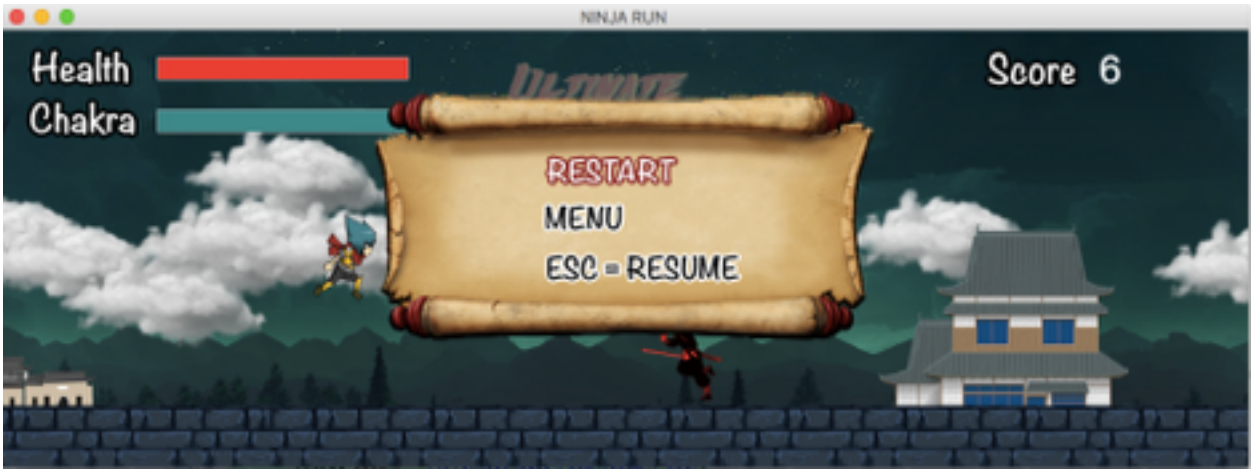

Figure 7: the Pause state

At Pause state, player can

- Press ESC to resume
- Press Enter to select option.
- Press Up/Down arrow to move option.

When health gauge is run out , ninja is died. The screen switch to game over state.

Figure 8: When ninja die

-Press Enter to go to game over state

This state will show your score and press enter to return menu.

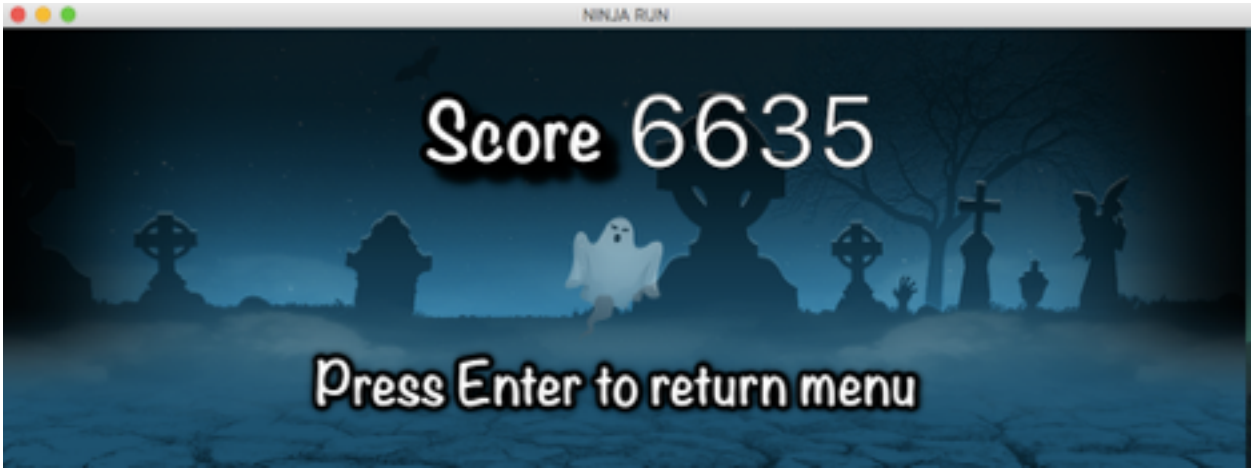

Figure 9 : Game over state

At game over state, player can

- Press Enter to return menu

2.Implementation Detail

Figure 8 : UML diagram of the program.

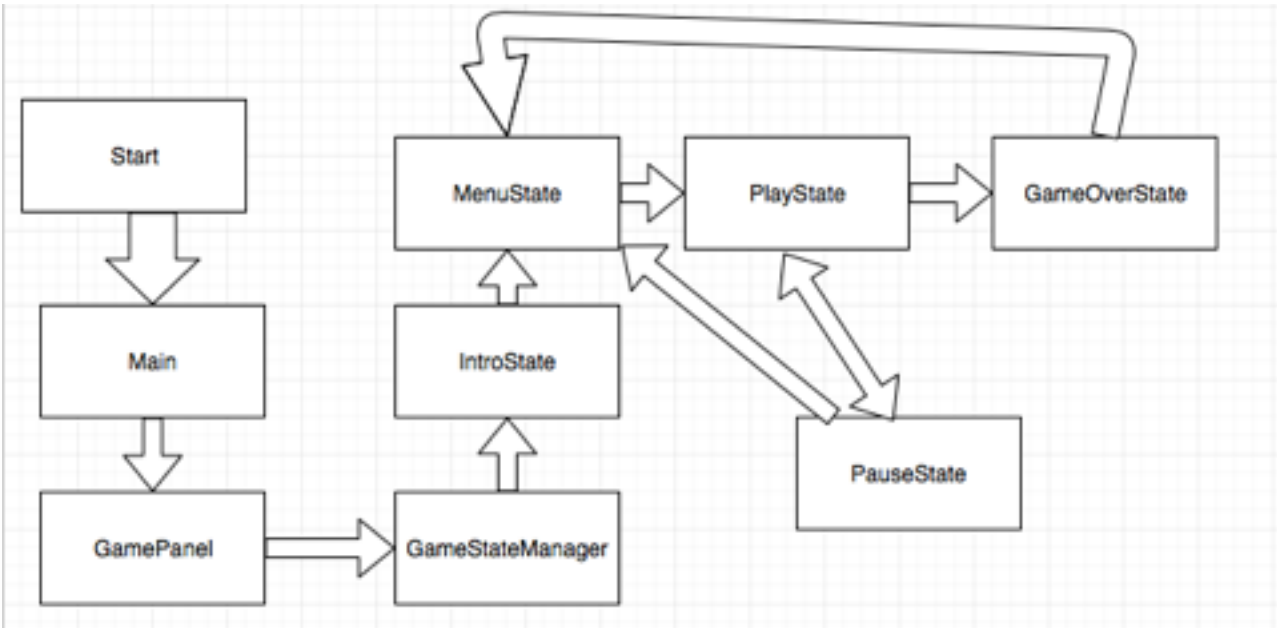

Figure 9 : The Relation of State class.

When the application starts ,Main call GamePanel and GamePanel call GameStateManager to draw the IntroState. So next state is MenuState ,It can stat game and exit game.When game is started PlayState draw a game , If we press ESC , It game pauses and shows pause screen. These can return to MenuState. When we lost game, State changes to GameOverState . It shows score and return MenuState

EveryState was control by GamePanel to update it every second and GameStateManager to switch state.

2.1 Package application

2.1.1 Class Main extends Application
2.1.1.1 Field

-Canvas panelCanvas	initialize GamePanel.
-Scene panelScene	initialize scene by GamePanal

2.1.1.2 Method

+void start(Stage primaryStage)	The main entry point for the JavaFX applications.
+void stop()	This method are going to execute before JavaFX application terminates. In this program, we use it to stop the remaining threads before exit the program.
+ void main(String[] args)	An entry point of the application.

-void addKeyEventHandler()	Add two KeyEvent handlers to the canvas. - When player press any letter key (A-Z) (KeyPressed), use characterInput to add pressed character. - When player release any letter key (A-Z) (KeyReleased), use CharacterInput to remove pressed character.

2.1.2 Class GamePanel extends Canvas implements Runnable
2.1.2.1 Field

-static Stage primaryStage	a primaryStage
+ static final int WIDTH	set width = 1000
+ static final int HEIGHT	set height = 350;
-Thread thread	A thread for game animation.
-static boolean running	Is game running
- int FPS	Number of frame rates per second. Default is 30.
-int TARGET_TIME	Time period between each update of a game animation ,Default is 1000/FPS
-GameStateManager	A game state manager.

2.1.2.2 Constructor

+GamePanel()	Initialize GamePanel,Set width = WIDTH , height =HEIGHT -requestFocus() -startAnimation()

2.1.2.3 Method

`+void startAnimation()`	Start the game animation loop thread and set Running to true.
`+void stopAnimation()`	set Running to false.
`+ void run()`	A game animation loop. -The loop will stop when Running is false. -The animation will update when the time has pass by a TARGET_TIME -For each loop, call update() and draw().
`-void init()`	Initialize GameStateManager. set Running to true
`-void update`	GameStateManager and Keys update
`-void draw`	GameStateManager draw in GamePanel.

2.2 Package SharedObject

2.2.1 Class RenderableHolder

This class is contain all resources for easier to call. We add all Image and Sound by create field then create a path in method ClassLoader and create assign an object to the field. This is some of the pattern we use it.

2.2.1.1 Field

+ static Image logo	A Logo image
+ static AudioClip gameplay	A gameplay audio
static loadResource()	call a method loadResource()

2.2.1.2 Method loadResource()

String Logo	An image path for logo by ClassLoader
String gamePlay	An audio path for gameplay by ClassLoader
logo = new Image(Logo)	
gameplay = new AudioClip(gameplay)	

2.3 Package Manager

2.3.1 Class GameStateManager

2.3.1.1 Field

- boolean pause	is game play pause
-PauseState pauseState	screen when state is pause
- static GameState[] gameState	list of state
+int currentState	present state
+int previousState	the state before
+int NUM_STATES	Total state ,set to 4
+int INTRO	set to 0
+int MENU	set to 1
+int PLAY	set to 2
+int GAMEOVER	set to 3

2.3.1.2 Constructor

+GameStateManagerl()	It is called when start program. -set pause to false -Initialize PauseState -Initialize list of GameState has NUM_STATES — index -set state to intro by method setState(

2.3.1.3 Method

+void setState(int state)	set previousState to currentState and set gameState of previousState to null by call unloadState() set state to state and that state call init()
+void setState(int state , int score)	set previousState to currentState and set gameState of previousState to null by call unloadState() set current State to GameOverState
+void unloadState (int state)	set state to null
+void setPaused(boolean b)	set pauses to b and init pauseState
+void update()	update gameState
+void draw(Canvas game)	draw the gameState
+ GameState getCurrentState()	return current state

2.3.2 Class Animation

2.3.2.1 Field

-List<Image> list	list of animation image
-long deltaTime	
-int currentFrame	
-long previousTime	

2.3.2.2 Constructor

+Animation(int deltaTime)	set deltaTime initialize List of image set preiousTime = 0

2.3.2.3 Method

+void updateFrame()	run animation on deltaTime ,go to next fame until end of list
+void resetFame()	set fame to 0
+void addFrame(Image image)	add image to list
+Image getFrame()	return currentFame

2.3.3 Class Keys

2.3.3.1 Field

+ static boolean int UP	set to 0
+static boolean int LEFT	set to 1
+ static boolean int DOWN	set to 2
+ static boolean int RIGHT	set to 3
+ static boolean int SPACE	set to 4
+ static boolean int ENTER	set to 5
+static boolean int ESCAPE	set to 6
+static final int NUM_KEYS	set to 8
+static boolean keyState[]	initial list of boolean ,Size is NUM_KEY
+static boolean prevkeyState[]	initial list of boolean ,Size is NUM_KEY

2.3.2.2 Method

+ void keySet(KeyEvent k , boolean b)	set state which has press to true
+void update()	set prevKeyState to keyState
+ boolean isPressed((int key)	return if keyState not equal to prevKeyState
+boolean isDown(int key)	return if key is pressed
+boolean anyKeyDown()	return if it has one in keyState pressed
+boolean anyKeypress()	return if it has one in keyState pressed and not equal previousKeyState

2.3.4 Class ObjectsManager

2.3.4.1 Field

- static final int MAX_CHURIKEN	set to 50
- Random rand	Random object to generate random integer
- int wait	For counting a time before a next wave of enemy
+ static int wave	Counting of wave
- List<Character> enemies	List of enemies
- List<Shuriken> shurikens	List of shurikens
- List<Effect> booms	List of smoke effect that appears when enemy die
- Ninja ninja	Ninja main character which running
- Heal heal	Special item in stage to increase health
- Speed speed	Special item in stage to unlock ultimate power
- int shurikencount	Counting of shurikens

2.3.4.2 Constructor

+ ObjectsManager(Ninja ninja)	Initialise ninja by given ninja Initialise all field Add first enemy to enemies by createEnemy(1) Set shurikencount to MAX_SHURIKEN Initialised heal at posX 4000 and speed at 1000

2.3.4.3 Method

+ void update()	Update enemies, shurikens and booms Check is shrines hit an enemies by checkfire() Update heal and speed If shuriken had fired increase shurikencount by 1
+ void draw(Canvas game)	Draw everything in order heal speed enemies shurikens booms
+ void boomupdate()	Return if no effect on screen Update every effect in booms then check if it's destroy remove it from list
+ void shurikenupdate()	Return if no shuriken on screen Update every shuriken in shurikens then check if it is out of screen remove it from list
+ void enemyupdate()	If no enemy on screen check if wait = 10 then spawn new wave of enemies Update all enemy then check if it is out of screen destroy it.

- Character createEnemy(int i)	Generate new gap between enemy start at pos 1000 then add 50 multiple i Random type of enemy If enemy type is FlyingObject add random(20) * 100 to the gap then return new FlyingObject If enemy type is Enemy add random(10)*100 to the gap then return new Enemy If enemy type is Obstruct return new Obstruct
+ void createShuriken()	Ninja cannot fire shuriken if he's in WARP state Make sure shurikencount is more than 0 Add new shuriken to shurikens with posX-10 and posY of ninja Play sound of shuriken Decrease shurikencount by 30
+ boolean isCollision()	Ninja will not collision with anything if he's in WARP or ULTIMATE state Check every Character in enemies if it collision with ninja then return true if no Character collision with ninja return false (Use intersects method for rectbound of ninja and Character))
+ void checkFire()	Return if no Character on screen Check only Character in Enemy type if it collision with shuriken if it is add new Effect to booms at that position then destroy enemy remove shuriken add play sound EnemyHitSound
+ boolean isSpCollision()	Like is Collision() but check with heal and speed then increase health or obtain power
+ void reset()	Remove all Character in enemies Create first character at (1) then add to enemies
+ int getWave()	Return wave
+ void increaseWave()	Increase wave by 1 If wave is more than 15 and less than 25 increase normal speed of ninja by 1 every 2 waves
+ void spawnEnemy()	Set wait to 0 Increase wave Add amount of Character to enemies equal to wave which add some generate random int
+ destroy(Character c)	Remove character c from enemies Up score for ninja

2.4 Package GameState

2.4.1 Class IntroState extends GameState
2.4.1.1 Field

-double alpha	color of logo that has fade
-int ticks	time to control logo to fade
- final int FADE_IN	time to fade in
- final int FADE_OUT	time to fade out
-LENGTH	time to show logo

2.4.1.2 Constructor

+ IntroState(GameStateManager gsm)	initial by GameStateManager

2.4.1.3 Method

+void init()	set ticks ot 0 play sound logo
+void update()	add ticks by 1 addKeyEventHandler -if ticks less than FADE_IN set alpha to 1-(1*ticks/FADE_IN) and if alpha <0 set alpha =0 -if tick in range FADE_IN to LENGTH set alpha to (1*ticks-FADE_IN-LENGTH)/FADE_OUT and if alpha>1 set alpha =1 -if tick more than FADE_IN+LENGTH+FADE_OUT set GameState to MenuState stop logo sound
+void draw(Canvas game)	draw Logo image and fad in and fad out by new color (0,0,0,alpha)
+addKeyEventHandler()	if press enter , skip to GameState change to MenuState
+static boolean prevkeyState[]	initial list of boolean ,Size is NUM_KEY

2.4.2 Abstract Class GameState extends Canvas

2.4.2.1 Field

protected GameStateManager gsm	

2.4.2.2 Constructor

+GameState(GameStateManager gsm)	set GameStatManager to gsm

2.4.2.3 Method

+void init()	start state
+void update()	update state
+void draw(Canvas game)	draw state
+void addKeyEventHandler()	Has event by press keyboard

2.4.3 Class MenuState extends GameState

2.4.3.1 Field

+boolean isKeyPressed	is key pressed
+Image bg	background image
- int currentOption	currentOption

2.4.3.2 Constructor

+MenuState(GameStateManager gsm)	initial GameStateManager

2.4.3.3 Method

+void init()	start state
+void update()	addKeyEventHandler
+void draw(Canvas game)	if currrentOption = 0 draw bg1 and land1 image if currentOption =1 draw bg2 and land1 to draw option start and quit men
+void addKeyEventHandler()	if press down current +=1 and currentOption = current%2 and play sound collect if press up current -=1 and currentOption =-1* current%2 and play sound collect if press enter to select option and play sound ninja
-void selectOption()	currentOption =0 : for to game PlayState currentOption =1 : for exit application

2.4.4 Class PlayState extends GameState

2.4.4.1 Field

- Land land	land
+int wave	wave of enemy
+Ninja ninja	ninja
- int START_GAME_STATE	set to 0
-int GAME_PLAYING_STATE	set to 1
-int GAME_OVER_STATE	set to 2
+int gameState	game state
- Font SCORE_TIME_FONT	initialize (“Monospace”,30)
-ObjectsManager objectsManager	ObjectsManager
- Clouds clouds	clouds
- BackgroundItem bgi	BackgroundItem
int coutinstruct = 0	count instruction in stat state to blink

2.4.4.2 Constructor

+PlayState(GameStateManager gsm)	initial GameStateManager set gameState to START_GAME_STATE initialize ninja ,land ,clouds,backgroundItem,objectManager

2.4.4.3 Method

+void init()	set speed ninja to 8
+void update()	addKeyEventHandler,countinstruct+=1 set wave = objectManager.wave CASE START_GAME_STATE -update land ninja clouds backgroundItem CASE GAME_PLAYING_STATE -update land ninja clouds backgroundItem objectsManager -if collision ,ninja take damage -if health <=0 ,ninja dead and set gameStateto GAME_OVER_STATE CASE GAME_OVER_STATE -update ninja

+void draw(Canvas game)	case START_GAME_STATE: -draw clouds , backgroundItem, ninja ,land and Instruction Press image . make Press image blink. case GAME_PLAY_STATE -draw clouds , backgroundItem, ninja ,land,objectManager -draw Health and Chakra and Score and Ultimate bar. case GAME_OVER_STATE -draw ninja land , objectManager and DeathBackground.
+void addKeyEventHandler()	case START_GAME_STATE: -if any key press , change gameState to GAME_PLAYING_STATE and set ninja start
- void resetGame	set ninja is not dead ninja reset objectManager reset

2.4.5 Class GameState extends GameState

2.4.5.1 Field

- int currentOption	current option

2.4.5.2 Constructor

+PauseState(GameStateManager gsm)	initial GameStatManager

2.4.5.3 Method

+void init()	set currentOption to 0
+void update()	addKeyEventHandler
+void draw(Canvas game)	draw background that “press esc to resume” if current ==0 draw pause scroll that select restart if current ==1 draw pause scroll that select menu
+void addKeyEventHandler()	if press down current +=1 and currentOption = current%2 and play sound collect if press up current -=1 and currentOption =-1* current%2 and play sound collect if press enter to select option if press esc to resume game
-void selectOption()	currentOption =0 : to game PlayState ,setPause to false currentOption =1 : to MenuState ,setPause to false

2.4.6 Class GameOverState extends GameState

2.4.6.1 Field

- int score	score
- Font TEXT_FONT	initial “Flipps” size 80
-int blink	blink constant

2.4.6.2 Constructor

+GameOverState(GameStateManager gsm , int score)	initial GameStatManager set score =score

2.4.6.3 Method

+void init()	
+void update()	addKeyEventHandler blink+=1
+void draw(Canvas game)	if blink %10 <6; draw game over ,ghost ,and score else draw game over which has massage,ghost ,and score
+void addKeyEventHandler()	if press enter to return MenuState

2.5 Package Object

2.5.1 Abstract class Character

2.5.1.1 Method

+void update()	update object
+void draw(Canvas game)	draw object
+Rectangle getBound()	get bond object
+boolean isOutOfScreen()	is object out of screen
-int getPosX()	return post
-int getPosY()	return posy

2.5.2 Class Obstruct extends Character

2.5.2.1 Field

+ static final int Y_LAND	Set to 300
- int posX	Position in X axis
- Image image	Image for obstruct

- Ninja ninja	Main character which playing
- Rectangle rectBound	Rectangle for this character

2.5.2.2 Constructor

Obstruct (Ninja ninja, int posX)	Initialise all field Initialise image from RenderableHolder.Spike

2.5.2.3 Method

+ void update()	Decrease posX by ninja's speed
+ void draw(Canvas game)	Draw image at posX and Y_LAND-image height
+ Rectangle getBound()	Set Rectangle Set X to posX+20 Set Y to Y_LAND- image's height Set width to image's width -10 Set height to image's height
+ boolean isOutOfScreen()	Return true if posX is less than negative of image's width
+ int getPosX()	Return posX
+ int getPosY()	Return Y_LAND

2.5.3 Class FlyingObject extends Character

2.5.3.1 Field

+ static final int Y_LAND	Set to 325
- double posX	Position in X axis
- int posY	Position in Y axis
- int width	Object's width
- int height	Object's height
- Animation badShuriken	Animation for object
- Rectangle rectBound	Rectangle for this character
- Ninja ninja	Main character which playing
- Random rand	Random object to generate random integer

2.5.3.2 Constructor

FlyingObject(Ninja ninja, int posX)	Initialise all field Initialise badShuriken from RenderableHolder.badShruiken initialise posY by random number between 0 and 15 then then multiply by 10 and add ninja height

2.5.3.3 Method

+ void update()	Decrease posX by ninja's speed * 2.5 Update animation frame Update width and height for avery frame
+ void draw(Canvas game)	Draw image at posX and Y_LAND-(height+posY)
+ Rectangle getBound()	Set Rectangle Set X to posX+10 Set Y to Y_LAND- height Set width to image's width -10 Set height to image's height
+ boolean isOutOfScreen()	Return true if posX is less than negative of image's width
+ int getPosX()	Return posX
+ int getPosY()	Return posY

2.5.4 Class Enemy extends Character

2.5.4.1 Field

+ static final int Y_LAND	Set to 325
- int posX	Position in X axis
- int posY	Position in Y axis
- Animation enemyAnim	Animation for enemy
- Ninja ninja	Main character which playing
- Rectangle rectBound	Rectangle for this character

2.5.4.2 Constructor

Enemy(Ninja ninja, int posX)	Initialise all field Initialise badShuriken from RenderableHolder.enemyAnim

2.5.4.3 Method

+ void update()	Decrease posX by ninja's speed *2 Update animation frame
+ void draw(Canvas game)	Draw image at posX and posY
+ Rectangle getBound()	Set Rectangle Set X to posX+10 Set Y to posY+5 Set width to image's width -10 Set height to image's height -10
+ boolean isOutOfScreen()	Return true if posX is less than negative of image's width
+ int getPosX()	Return posX

+ int getPosY()	Return Y_LAND

2.5.5 Class Shuriken extends Character

2.5.5.1 Field

- float posX	Position in X axis
- float posY	Position in Y axis
- Animation shuAnim	Animation for shuriken
- int width	Object's width
- int height	Object's height
- Rectangle rectBound	Rectangle for this character
- Random rand	Random object to generate random integer

2.5.5.2 Constructor

+ Shuriken (int posX, int posY)	Initialise all field Initialise shuAnim by 10 delta time Generate random integer to choose between frame Kunai or Shuriken then add it to shuAnim

2.5.5.3 Method

+ void update(int currentSpeed)	Overload method update with integer to increase posX by 20 and currentSpeed Update animation frame
+ void draw(Canvas game)	Draw animation's frame at posX and posY
+ Rectangle getBound()	Set Rectangle Set X to posX+10 Set Y to posY+5 Set width to image's width*1.25 Set height to image's height*1.25
+ boolean isOutOfScreen()	Return true if posX more than 1000
+ int getPosX()	Return posX
+ int getPosY()	Return posY

2.5.6 Class Heal extends Character

2.5.6.1 Field

- int posX	Position in X axis
- int posY	Position in Y axis at 200
- Image image	Image for Healing object
- int width	Object's width
- int height	Object's height

- Ninja ninja	Main character which playing
- Rectangle rectBound	Rectangle for this character

2.5.6.2 Constructor

+ Heal (Ninja ninja, int posX)	Initialise all field Initialise image from RenderableHolder.Heal Width and Height are minus by 10

2.5.6.3 Method

+ void update(int currentSpeed)	Overload method update with integer to increase posX by 20 and currentSpeed Update animation frame
+ void draw(Canvas game)	Draw animation's frame at posX and posY
+ Rectangle getBound()	Set Rectangle Set X to posX Set Y to posY Set width to image's width*1.2 Set height to image's height*1.2
+ boolean isOutOfScreen()	Return true if posX less than negative of image's width then reset object
+ void reset()	Generate random posX and posY for new Heal
+ int getPosX()	Return posX
+ int getPosY()	Return posY

2.5.7 Class Speed extends Character

2.5.7.1 Field

- int posX	Position in X axis
- int posY	Position in Y axis at 200
- Image image	Image for Speed object
- int width	Object's width
- int height	Object's height
- Ninja ninja	Main character which playing
- Rectangle rectBound	Rectangle for this character

2.5.7.2 Constructor

+ Speed (Ninja ninja, int posX)	Initialise all field Initialise image from RenderableHolder.Speed Width and Height are minus by 10

2.5.7.3 Method

+ void update(int currentSpeed)	Overload method update with integer to increase posX by 20 and currentSpeed Update animation frame
+ void draw(Canvas game)	Draw animation's frame at posX and posY
+ Rectangle getBound()	Set Rectangle Set X to posX Set Y to posY Set width to image's width*1.2 Set height to image's height*1.2
+ boolean isOutOfScreen()	Return true if posX less than negative of image's width then reset object
+ void reset()	Generate random posX and posY for new Speed
+ int getPosX()	Return posX
+ int getPosY()	Return posY

2.5.8 ClassEffect

2.5.8.1 Field

+static final int DURATION	time of animation ,Set to 10
+int posX	position fo in x axis
+int posY	position fo in y axis
+float speedX	speed in x axis
+int count	count
-Animation boomAnim	animation of effect boom

2.5.8.2 Constructor

+Effect(int posX , int posY, float speedX)	set posX , posY ,speedX to the parameter set count to 0 initial boomAnim

2.5.8.3 Method

+ void update()	posX-=speedX boomAnim update add count by 1
+void draw(Canvas game)	if count less than DURATION , draw boomAnim in posX and posY
+ boolean isDestroy	if count >= DURATION return true

2.5.9Class Land

2.5.9.1 Field

+static final int LAND_POSY	position of land ,set to 300
-static List<ImageLand> listland	list of land
-Image land1	image of land

2.5.9.2 Constructor

+Land(int width , Ninja ninja)	initial land1 , listLand set ninja to a parameter create 3 ImageLand and make it continue.

2.5.9.3 Method

+ void update()	posX of current land -= speed of ninja. if island has next item ,set posX of next land to previous posX + land width. if current land is out of screen , remove it and add it in back of list
+void draw(Canvas game)	draw every land
-void setImageLand(ImageLand imgLand)	image = land1
- class ImageLand	has field : float posX Image image

2.5.10 Class BackgroundItem

2.5.10.1 Field

- List(Image item) listitem	list of item
-Image item1	first item
-Image item2	second item

2.5.10.2 Constructor

+BackgroundItem(int width, Ninja ninja)	set ninja . initialize item1,item2,listitem create 2 Imageitem fist posX=100 second posX =900

2.5.10.3 Method

+ void update()	posX of first -=speed of ninja /4 if list item has next item , posX of it-=speed of ninja /4 if item out if screen , remove it and set posX =100 and add it in back of list
+void draw(Canvas game)	draw every in list item
-private class Imageitem	Image item float posX=100 int posY=155

2.5.11 Class Clouds

2.5.11.1 Field

- List(ImageCloud) listCloud	list of Clouds
-Ninja ninja	ninja
-Image cloud2	image of cloud

2.5.11.2 Constructor

`+Clouds(int width, Ninja ninja)`	set ninja . initialize cloud2,listCloud create 10 ImageCloud (posX,posy):1 (0,100) 2.(50,120),3(125,150)4,(175,110)5.(200,100)6.(225,50)7.(275,110)8.(500,50)9.(530,125)10.(570,50)11.(750,100)12(1000,125) and add to list

2.5.11.3 Method

+ void update()	posX of first -=speed of ninja /8 if listCloud has next item , posX of it-=speed of ninja /9 if item out if screen , remove it and set posX =100 and add it in back of list
+void draw(Canvas game)	draw every clouds in list
-private class ImageCloud	Image item float posX=1000 int posY=100

2.5.12 Class Ninja

2.5.12.1 Field

- static final float MAX_SPEED	Set to 25
- static final int LAND_POSY	Set to 220
- static final float GRAVITY	Set to 0.98f
- static final int MAX_HEALTH	Set to 5
- static final int HIT_COOL_DOWN	Set to 20
- static final int WARP_COOL_DOWN	Set to 10
- static final int WARP_TIME	Set to 25
- static final int ULTIMATE_TIME	Set to 74
+ static final int NORMAL_RUN	Set to 0
+ static final int JUMPING	Set to 1
+ static final int COOLDOWN_RUN	Set to 2
+ static final int COOLDOWN_JUMP	Set to 3
+ static final int DOWN_RUN	Set to 4
+ static final int DEATH	Set to 5
+ static final int WARP	Set to 6
+ static final int ULTIMATE	Set to 7
+ boolean isStart	Contain boolean is it start playing
+ int coolDown	Hit cool down count
+ int warpCoolDown	Warp cool down count
+ int timewarp	Warp duration count
- float posX	Position in X axis
- float posY	Position in Y axis
- float speedX	Current running speed
- Rectangle rectBound	Rectangle for this character
- int jumpcount	Jumping count
- int warpcount	Warp count
- int health	Current health
- int currentstate	Current state

- int powerobtain	Current power to unlock ultimate
- int ultimateTime	Time counting for ultimate duration
- int ghost	Ghost time counting in DEATH state
- Animation normalRunAnim	Animation for NORMAL_RUN and DOWN_RUN states
- Animation cooldownRunAnim	Animation for COOLDOWN_RUN state
- Animation cooldownJumpAnim	Animation for COOLDOWN_JUMP state
- Animation warping	Animation for WARP state
- Image jumping	Image for JUMPING state
+ int score	Current score
- int count	Counting for anything

2.5.12.2 Constructor

+ Ninja()	Initialise all field Set currentstate to NORMAL_RUN Set ultimateTime to ULTIMATE_TIME set isStart to false

2.5.12.3 Method

+ floaet getSpeedX()	Return speedX
+ void setSpeedX(float speedX)	Set speedX and normalspeedX to input speed make sure it not over MAX_SPEED
+ void draw(Canvas game)	Draw an animation or image form each stage on canvas
+ void update()	If it not start yet update only noemalRunAnim - If it is DEATH state increase ghost and posY - Update score - Decrease wapCoolDown - Set state to the right state from data - Update each state in proper way
+ void jump()	Make sure state is only run and jump You can only jump twice per time When jump set speedY to -20f then increase jump count and increase posY by speedY and set state to JUMPING
+ void down()	If you in the air this method will pull you down to the ground , Otherwise it will increase your speed (sprint) set state to DOWN_RUN

+ void warp()	You can only warp once until you touch the ground Save state to currentstate then change state to WARP and set warpCoolDown to WARP_COOL_DOWN and play sound
+ void ultimate()	Make sure you already unlock ultimate power Set state to ULTIMATE then play sound
+ Rectangle getBound()	Set X to posX+5 Set Y to posY Set width to image's width-10 Set height to image's height
+ int getHeight()	Return height of current frame
+ void dead(boolean isDeath)	Set state to DEATH then play sound
+ void reset()	Set posY to LAND_POSY Reset health and power then set isStart to false
+ void updateScore()	Make sure it already start then increase score every 10 count
+ void upScore()	Increase score by 20
+ void takeDamage()	You will be invisible in WARP and ULTIMATE state If your cool down time is out decreaseHealth and set coolDown to HIT_COOL_DOWN and play sound
+ void decreaseHealth()	Decrease health by 1
+ void increaseHealth()	Make sure your health is less than MAX_HEALTH then increase it
+ int getHealth	Return Health
+ void resetHealth()	Set health to MAX_HEALTH
+ void powerObtain()	Check powerboat is less than 3 Increase powerobtain by 1
+ boolean isUltimateReady()	Check if powerboat more than 3 return true Otherwise return false
+ boolean isOutOfScreen()	Return false
+ void int getState()	Return state
+ void setState(int state)	Set state to given state
+ float get posX()	Return posX
+ float get posY()	Return posY
+ void increaseNormalSpeedX(int i)	increase normalspeedX by 1